Recommendation based on Mining Product Reviewers' Preference Similarity Network

Feng Wang and Li Chen
Department of Computer Science
Hong Kong Baptist University, Hong Kong
{fwang, lichen}@comp.hkbu.edu.hk

ABSTRACT

Most products in e-commerce are with high cost (e.g., digital cameras, computers) and hence less likely experienced by users. Thus, the traditional recommender techniques (such as user-based collaborative filtering and content-based methods) are not applicable, because they largely assume that the users have prior experiences with the items. The "new user" is hence a typical phenomenon and challenging issue that recommender systems face in this environment. In this paper, we have particularly proposed to build product reviewers' preference similarity network to solve this problem. Specifically, with product reviews, we have attempted to recover each reviewer's weight preferences over features, based on which all reviewers can be connected in an implicit network with the edge denoting their pairwise preference similarity. We additionally adopted the *Latent Class Regression Model* (LCRM) to identify the sub-communities in this network, where each sub-community corresponds to a cluster of like-minded reviewers. The new user's stated feature preferences can be then matched to the cluster with most relevant reviewers and their reviewed products can be taken as recommendation candidates. The experimental results reveal that our novel method outperforms others that either did not consider reviews, or did not attempt to reconstruct reviewers' inherent feature preferences (from their written reviews) for benefiting the recommendation process. The LCRM-based clustering method was also proven with higher accuracy than related ones (like k-Means based clustering), especially when the new user's stated preferences were less complete.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Retrieval and Search—*Information Filtering*

General Terms

Algorithms, Experimentation

Keywords

Recommender system, inexperienced products, product reviews, weighted feature preferences, Latent Class Regression Model, similarity network

1. INTRODUCTION

In many of user communities exiting online, one special community can be called reviewers' community where the reviewer refers to the person who has expressed opinions on the product that s/he purchased or used. As a matter of fact, such network popularly appears in e-commerce website and the info provided by reviewers is helpful for a new buyer to judge product quality and make more confident purchase decision [5]. Though those reviewers are not explicitly socially connected, if we could built a network to relate them and use the edge between every two reviewers to show their inherent preference similarity, such *implicit network* could be potentially beneficial to enhance the recommender's power especially in inexperienced product domains (e.g., digital cameras, computers) for addressing the typical "new users" and "cold-start" problems. Indeed, because in high-risk and inexperienced product domains, users usually do not have much prior usage and/or purchase experiences before a new purchase, the standard recommender technologies, such as the collaborative filtering technique [4], the content-based method [3], and matrix factorization approaches [16], are not so directly applicable since they all stand on the fact that there are adequate user records (such as ratings) for them to predict the user's preferences over un-rated items. That is why in this paper we have particularly studied how to effectively incorporate reviewers' implicit network into the process of generating recommendations for new buyers.

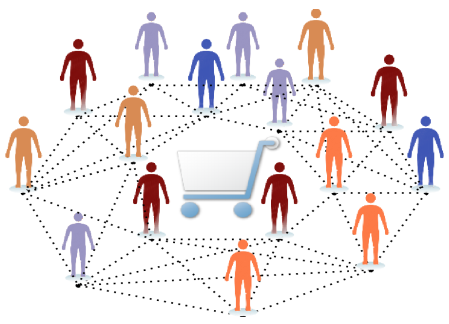

Figure 1: The reviewers' implicit preference similarity network.

The three specific tasks that we have exerted to perform are: 1) to first build reviewers' *implicit, similarity network* where every node represents a reviewer and every edge denotes two reviewers' preferences similarity (see Figure 1). We further propose an approach to cluster reviewers into sub-communities so that every sub-community contains like-minded reviewers according to their intrinsic preference similarity over product features. 2) To elicit the new buyer's preferences on features, based on which we could find a cluster of reviewers that is most relevant to the new buyer. 3) The products highly praised by these located reviewers can be then taken as recommendation candidates. A prediction score is computed for each product to indicate its satisfaction degree.

For the first task, though in most applications reviewers provided "overall rating" along with their textual comments on the products, it is not so straightforward to conduct pure rating-based similarity measure, given that every reviewer comments on only one or few products (see our data analysis in Section 3). Therefore, it should be potentially more accurate if the similarity can be identified based on reviewers' intrinsic preferences over product features (such as "image quality", "battery life", "ease of use" of cameras). That is why in this step we have attempted to recover their feature preferences by analyzing their un-structured textual reviews. Although recent opinion mining systems can extract features and their associated opinions from text reviews [13], they are limited to further build on these opinion outcomes to reveal reviewers' preferences. Essentially, a reviewer's preferences can contain the importance degrees that s/he placed on respective features (so called the feature's weight). In Section 4.1.2, we will in detail describe how we have adopted *probabilistic regression model* to reveal such preferences, by considering three major elements: the reviewer's opinion values on features (as retrieved from reviews), the occurring frequency of features in reviews and the overall rating for products. Furthermore, we integrate the Latent Class Regression Model (LCRM) to identify inter-reviewer similarity and form clusters automatically. Relative to K-means [18], the LCRM was believed to have theoretical advantage and has been widely applied to conduct market segmentation [26], but this method was rarely investigated in the area of recommender system. Thus, in the experiment part, we have empirically compared the two clustering methods and identified LCRM's practical merit in representing reviewers' preference heterogeneity.

For the second task, we can ground on our prior work on preference elicitation to obtain the new user's feature preferences. Specifically, previously we have proposed the critiquing agent with the primary goal of stimulating users to state and refine their preferences on features [8]. The interaction normally continues several cycles and during every cycle the system presents multiple example products for the user to critique (so the method is also called *example critiquing*). The critiques reflect the user's criteria on improving some specific feature values and the agreement to compromise other less important ones. The critiquing action is hence naturally in accordance with the tradeoff navigation, which is a strategy strongly encouraged by decision theory to enable high-quality decision [21]. Thus, from the user's critiques, we can infer the importance degrees that s/he places on different features. However, previous series of user studies also indicated that though the user's preferences are feasibly elicited, due to her/his incomplete product knowledge, the stated preferences are usually partial (i.e., on a subset of features) [7, 8, 21]. The traditional preference-based ranking method based on the multi-attribute utility theory (MAUT) [15] is therefore limited since the returned products are purely matched to the user's stated preferences. The approach proposed in this paper is hence targeted to compensate for the previous work's limitation, by means of integrating reviewers' preferences, so that the active buyer's preferences over all features might be inferred based on reviewers.

Once we have the reviewers' similarity network built and the new buyer's (partial) preferences elicited, we could adopt the classical collaboration filtering mechanism to compute recommendation (the third task). Basically, every product as commented by at least a reviewer (who belongs to the most relevant cluster to the buyer) is computed with a prediction score via the weighted sum of reviewers' overall ratings on that product, where the weight is the preference similarity between a reviewer and the buyer (see details in Section 4.2.2). The top-N products with higher scores will be then presented to the buyer as the recommendation set.

In order to assess whether the recommendation set, being resulted from the above steps, likely contains the new buyer's target choice (i.e., the ideal product for the buyer to purchase), we experimentally compared it to four related approaches (see Section 5.2). The experiment demonstrates our method's superior recommendation accuracy, particularly owing to its process of building reviewers' similarity network and conducting LCRM-based auto-clustering. Especially, our method outperforms others when the new buyer's stated preferences are less complete, which hence indicates that it can be effectively integrated with the critiquing agent to generate recommendations once the critiquing agent elicits the user's preferences.

2. RELATED WORK

The works most relevant to ours can be classified into two branches: one is commonly termed as *multi-criteria recommenders*, with the primary goal of addressing the single-rating induced limitations; and another can be categorized as *review-based recommenders* because they explicitly incorporate reviews into the recommendation process. In the following, we introduce the state-of-the-art and indicate their inherent limitations.

The traditional recommender approaches, such as collaborative filtering (CF) and content-based ones [3, 4, 16], only consider users' single ratings on items, which however can not reveal why users gave such ratings. For example, two users rated a movie with the same rating but of different reasons: one likes its genre and director, and another likes its actor and actress. Thus, more works in recent years have attempted to reveal users' ratings on multi-facets of an item and developed the so called *multi-criteria based recommenders*. For instance, in [2], classic collaborative filtering was extended by utilizing user-stated multi-criteria ratings, for calculating user-user similarity. They also proposed to learn an overall rating via the aggregation function on multi-criteria ratings. This approach achieved better performance compared to traditional single-rating based CF method. In [20], they aimed to identify the dependency structure between the overall rating and multi-criteria ratings and utilized flexible mixture model to predict the rating for un-known items to a user. [17] adopted the additive utility analysis to estimate the utility of an item by integrating the marginal utilities of a user's multiple criteria on the item's attributes. It can hence be seen that the common objective of these methods was to estimate whether a user would be interested in an un-known item based on her/his multi-criteria ratings on known items. However, these works were largely targeted to recommend items to repeated users, so they have been mainly applied in low-risk and experienced product domains. They are therefore unfit for the cases with few ratings obtained from individual users (i.e., for inexperienced products), and inapplicable to handle the "new user" challenge. In our work, though we have also strengthened multi-criteria, we particularly propose to infer consumers' multi-criteria from their written reviews, and emphasize on utilizing these inferred *weight preferences on features* to generate recommendations to new users.

Another related branch of work has taken product reviews into account to offer product recommendations, but the main focus was simply on enhancing traditional CF methods via deriving one-dimensional virtual ratings from reviews' sentiment classification results [22, 29]. To the best of our knowledge, few works have in-depth explored the effect of multi-dimensional feature-level sentiments (i.e., opinions associated various features as contained in the reviews) on enhancing recommenders. In [14], they proposed a multi-relational matrix factorization (MRMF) method, which is an extension to low-norm matrix factorization, to model the correlations among users, movies and the opinions regarding specific features. This method however was still mainly oriented to experienced product domains. Another work which is more related to ours is [1], which adopted the feature-level sentimental results to enrich the cameras' description, based on which the product ranking was conduced. However, they neither evaluated the algorithm's accuracy, nor explored other possibilities, like the integration of inter-reviewer similarity into recommendation.

To summarize, the limitations of related works are mainly at the following aspects: 1) they are mostly oriented to serve low-risk, experienced product domains, and less addressed the issue of "new buyers" when the users search for inexperienced products. 2) The feature-level review analysis was rarely considered to support recommendations. Little work has either built on the review mining outcomes to infer reviewers' feature preferences and establish their inherent similarity network. 3) Related review-based recommenders such as [1] did not experimentally test their systems' recommendation accuracy when users' stated preferences were incomplete.

3. DATA ANALYSIS

In this section, we first give some statistic analysis on a real-world dataset. This dataset was gathered from `www.buzzillions.com`, with 7485 digital camera reviews to 186 digital cameras. The mean number of reviews to each product is 40, and the standard deviation (St.d.) is 31.56. These reviews were posted by 3754 reviewers in total. Fig. 2 shows the distribution among these reviewers in respect of the number of products they wrote review(s). It can be seen that about 69.80% of reviewers only commented one product, and the average number of reviews per reviewer is 1.99. Moreover, every review on average mentioned 4.7 distinct features (St.d. = 1.91).

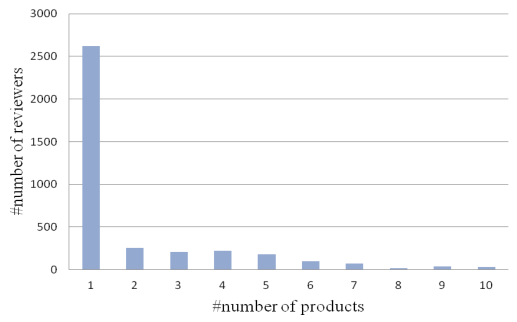

Figure 2: Distribution of the number of products per reviewer.

These preliminary analysis suggests that in the high-risk, inexperienced product domains, the number of products that each reviewer commented is limited given that s/he might have purchased or used few products. Thus, it is ineffective to compute two reviewers' preference similarity purely based on the overall ratings that they gave to the products. This makes most of existing recommender techniques unapplicable in such environment. Another finding, on the other hand, suggests that the adequate number of features (and associated opinions) as indicated in reviews might be more likely to help induce reviewers' preferences and establish their similarity network. Thus, in our work, we have emphasized inferring reviewers' weighted feature preferences and building inter-reviewer similarity link based on their reviews.

4. PROBLEM STATEMENT AND OUR METHODOLOGY

More specifically, we have been engaged in addressing the following three research questions:

1. ***How to recover the reviewer's weight preferences over features from her/his written reviews?*** Here, we assume every reviewer inherently has a weighted preference model, which is formally denoted as: $Pref_u = \{< f_i, w_{ui} > | 1 \leq i \leq n\}$, where w_{ui} indicates the importance degree that the user u places on feature f_i. The preference structure is theoretically grounded on Multi-Attribute Utility Theory (MAUT) [15] because it can explicitly consider trade-offs among attributes (i.e., features in our term) via the *weights*.

2. ***How to build reviewers' preference similarity network?*** When we have inferred the reviewers' feature preferences, it will be feasible to build a similarity network where each node denotes a reviewer and each edge represents the implicit relation between two reviewers as indicated by their preference similarity. Furthermore, we apply clustering technique to divide this network into sub-communities (i.e., clusters in our term) so that reviewers who belong to the same sub-community can be of tight preference-closeness.

3. ***How to leverage such reviewers' preferences into computing recommendation list so that the list can likely contain the new user's target choice?*** In order to effectively utilize the reviewers' preference similarity network, we propose associating the new

user to a cluster of reviewers according to their preference relevance. These reviewers' preferences on features that the new user has not stated might also help predict the new user's full preferences and locate recommendable products that s/he truly likes.

To illustrate the motivation behind our approach, let us first consider the following two real review examples respectively given by A and B (one to digital camera $C1$ and another to $C2$):

EXAMPLE 1. Reviewer A wrote a review to camera $C1$ ($overall\,rating = 5$).

"It can produce a great image in low light environment. You can usually use it in AUTO mode and expect a good result. If you don't mid a little bit heavier and bigger camera compared with most of compact cameras, this is the one you should get it. Only con I can think of is its little bit short battery life. Better to consider to buy an additional battery."

EXAMPLE 2. Reviewer B wrote a review to camera $C2$ ($overall\,rating = 5$).

"Takes great pictures. Best pictures I've seen from indoors with lower light. Slips easily into your pocket. Pity the settings dial is in the place you expect the button to take the picture. Also, the battery life is not great so I'd get a spare. There is a deal for both. Overall, I'm very impressed and delighted with my purchase"

In fact, either of the two reviewers only wrote one review as shown above. It then comes to the issue of how to infer their preference similarity. From the examples, it can be seen that both reviewers expressed positive opinion on the feature "image" and negative opinion on the feature "battery life". However, it is not straightforward to suggest that they have similar preferences since the two reviewed products are different. Therefore, one feasible solution is to infer the importance degree that the reviewer placed on the feature, named as *weighted feature preference*. A simple action for this purpose might count the frequency of feature that occurs in the review, under the intuition that higher occurrence of a feature might indicate that it has greater influence on the reviewer's overall rating (so being more important to her/him). But the weakness of this approach is that: 1) it cannot distinguish features with equal amount of occurrences. For example, in the example 2, the features "image" (using a synonymous term "pictures") and "battery life" apparently have different weights (as the "battery life" feature's opinion and the "image" feature's opinion have different correlation degrees with the overall rating) even though they are with equal amount of occurrences; 2) In some cases more frequent features are actually less important. For example, in the example 1, the feature "battery life" occurred more than the feature "image", but it can be derived from this review that it is less important than "image" to the reviewer (as the "battery life" feature' opinion has opposite polarity with the overall rating).

Thus, the critical part in our approach is to recover the reviewers' weighted feature preferences based on their written reviews. Specifically, we take into account the correlation between the overall rating and the opinion values of features (as extracted from reviews) to make the inference. The occurrence of feature is primarily taken as a type of prior knowledge being integrated into the model.

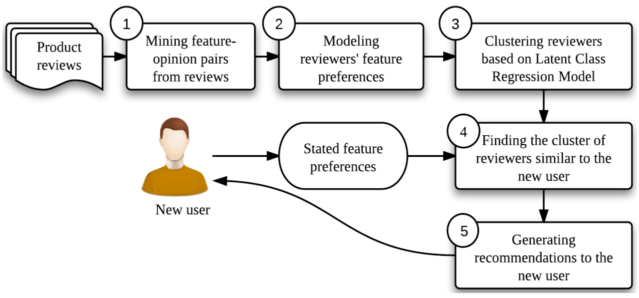
Product reviews
1
Mining feature-opinion pairs from reviews
2
Modeling reviewers' feature preferences
3
Clustering reviewers based on Latent Class Regression Model
New user
Stated feature preferences
4
Finding the cluster of reviewers similar to the new user
5
Generating recommendations to the new user

Figure 3: The workflow of our system.

The work flow of our system is concretely presented in Figure 3 which consists of five major processing steps: 1) conducting feature-level opinion mining to identify the pairs of *features* and *opinions* from individual reviews, for which the synonymous features were grouped and the opinion was quantified in the range of $[1, 5]$; 2) adopting a probabilistic regression model to derive the reviewer's weighted preferences on features based on the outcomes of Step 1); 3) applying the Latent Class Regression Model (LCRM) to cluster reviewers according to their preferences; 4) identifying a cluster of reviewers that is most relevant to the new user; and 5) generating recommendations according to these reviewers' commented products. Table 1 summarizes the notations used throughout the paper.

Table 1: Notations in this paper

Notation	*Description*
$REV = \{rev_1, \ldots, rev_M\}$	A set of M reviewers.
$\mathcal{P} = \{p_1, \ldots, p_{\lvert\mathcal{P}\rvert}\}$	A set of $\lvert\mathcal{P}\rvert$ products.
$\mathcal{S} \subseteq REV \times \mathcal{P}$	A set of reviewer-product pairs, where $(rev_i, p_j) \in S$ indicates that a reviewer rev_i wrote review to a product p_j.
$\mathcal{F} = \{f_1, \ldots, f_n\}$	The n features extracted from reviews.
r_{ij}	The review written by reviewer rev_i regarding product p_j.
R_{ij}	The overall rating reviewer rev_i gave to product p_j.
$\boldsymbol{X}_{ij} = [x_{ij1}, \ldots, x_{ijn}]$	The opinion values on the set $\mathcal{F}$ as extracted from a review r_{ij}.
$\boldsymbol{W}_{rev_i} = [w_{i1}, \ldots, w_{in}]$	The reviewer rev_i's weighted preferences, where each w_{il} is the weight on feature $f_l \in \mathcal{F}$, which could be None if the reviewer did not express any opinions on that feature.
$\mathcal{C} = \{c_1, \ldots, c_K\}$	The K clusters of reviewers.

4.1 Recovering Reviewers' Weighted Feature Preferences

For reviewers who already bought products, our objective was to infer and rebuild their weighted feature preferences from their reviews to the products. The basic idea behind our approach is that the overall rating of each reviewer can be considered as the weighted sum of her/his opinions on features, based on which we could learn the reviewer's *weighted preferences*. So this section is divided into two sub-steps: 1) mining feature-opinion pairs from product reviews; 2) modeling reviewers' weighted feature preferences.

4.1.1 Mining Feature-Opinion Pairs from Product Reviews

In the past decade, some effort has been devoted to conducting document-level or feature-level opinion mining (or called sentimental classification and sentimental analysis respectively) in the areas of natural language processing and data mining [13, 24]. Particularly, the *feature-level* opinion mining can return a set of $< feature, opinion >$ pairs from a review, where opinion indicates positive, neutral, or negative sentiment that a reviewer expressed on the feature. Therefore, our work can be regarded as the extension to their work, with the emphasis on refining the sentiment analysis results and further exploiting them to derive a reviewer's feature preferences on products.

We first used a Part-of-Speech (POS) tagger to extract the frequent nouns or noun phrases from reviews, which are the prospective feature candidates. Moreover, considering that reviewers often use different words or phrases to refer to the same product feature (e.g., "picture", "image" and "appearance"), we manually defined a set of seed words and grouped the synonymous features by computing their lexical similarity to the seed words. The lexical similarity is concretely determined via WordNet [11].

We then extracted opinions as associated to each feature in a review sentence. Most of existing works depended on the co-occurrence of product features and opinion bearing words for this purpose [13, 23]. However, these methods cannot identify opinions that are not so "close to" the feature. Therefore, we took advantage of a syntactic dependency parser [1], because it can return the syntactic dependency relations between words in a sentence. For example, after parsing the sentence "*It takes great photos and was easy to learn how to use*", "great" is identified with dependency relation AMOD with "photos" (meaning that "great" is an adjectival modifier of the noun word "photos"), and *"easy"* has COMP dependency relation with *"learn"* (indicating that "easy" is an open clausal complement of "learn"). In another example "*The photos are great*", "great" has NSUBJ relation with "photos" (indicating that "photos" is the subjective of "great"). Thus, we took all the words with such relations with the product feature words as opinions.

After identifying the *feature-opinion pairs* from a review sentence, the next task was to assess the opinion's sentiment strength (also called polarity). For this goal, we applied SentiWordNet [10] because it provides us with a triple of polarity scores (i.e., positivity, negativity and objectivity, respectively denoted as $\{Pos(s), Neg(s), Obj(s)\}$, and each ranging from 0.0 to 1.0. $Pos(s) + Neg(s) + Obj(s) = 1$), for each opinion word. The triple scores are then merged into a single sentiment value for the opinion word s: $O_s = Neg(s) * R_{min} + Pos(s) * R_{max} + Obj(s) * \frac{R_{min}+R_{max}}{2}$ (where R_{min} and R_{max} represent the minimize and maximal rating scales respectively. We set them as $R_{min} = 1$, $R_{max} = 5$ so that O_s ranges from 1 to 5). If there are negation words (e.g., not, don't, no, didn't) in a sentence, the polarity of related opinions is reversed.

We then aggregated all opinion words' sentiment values respecting a specific feature in a review. Instead of using the simple arithmetic mean as commonly made by others, we performed a weighted average by which each opinion word's sentiment value also behaves as a weight, so that the extremely positive or negative polarizations are less susceptible to shift. For instance, if two opinion words, "good" and "great" are associated to a feature, the feature's final opinion value is hence ($\frac{4\times4+5\times5}{4+5} = 4.55$) where 4 and 5 are the sentiment values of the two words ("good" and "great") respectively. And for a absence feature not involved a review, its opinion value will be set as the default neural value (3 in our case).

4.1.2 Modeling Individual Reviewer's Weighted Feature Preferences

The next step was then to derive the reviewer rev_i's weighted preferences $\boldsymbol{W}_{rev_i}$ on product features $\mathcal{F}$. To achieve this goal, we applied *probabilistic regression model* for learning the weights [28]. Specifically, given features' opinion values $\boldsymbol{X}_{ij} \in \mathbb{R}^n$ in respect of a product p_j as given by the reviewer rev_i (resulted from the above step), her/his overall rating R_{ij} can be drawn from a Gaussian distribution around $\boldsymbol{W}_{rev_i}^T \boldsymbol{X}_{ij}$:

$$Pro(R_{ij}|\,\boldsymbol{W}_{rev_i}, \boldsymbol{X}_{ij}, \sigma^2) = \mathcal{N}(R_{ij}|\,\boldsymbol{W}_{rev_i}^T \boldsymbol{X}_{ij}, \sigma^2) \quad (1)$$

where $\boldsymbol{W}_{rev_i}$ can be formally represented by a Multivariate Gaussian Distribution, $\boldsymbol{W}_{rev_i} \sim \mathcal{N}(\mu, \Sigma)$, with μ as the mean and Σ as the covariance matrix. The reason behind using Gaussian regression model is because it is suitable to model consumers' preferences given prior knowledge (like the frequency of features) according to Conjoint Analysis in [9].

We additionally incorporated the occurrence frequency of a feature in a review (denoted as μ_0) into the model, being the prior knowledge of μ. It can be essentially used to define the distributions of μ and Σ based on its Kullback-Leibler (KL) divergence to the prior distribution with mean μ_0 and identity covariance matrix I:

$$Pro(\mu, \Sigma) = \exp[-\psi \cdot KL(Q(\mu, \Sigma)||Q(\mu_0, I))] \quad (2)$$

where $KL(\cdot, \cdot)$ is the KL divergence, $Q(\mu, \Sigma)$ denotes a multivariate gaussian distribution, and ψ is a tradeoff parameter ($\psi = 100$ in our experiment).

The probability that a overall rating R_{ij} is given to a product p_j (that accompanies a review r_{ij}) can be hence like:

$$\begin{aligned} Pro(R_{ij}|\Psi, r_{ij}) = \int (&Pro(R_{ij}|\,\boldsymbol{W}_{rev_i}, \boldsymbol{X}_{ij}, \sigma^2) \\ &\cdot Pro(\,\boldsymbol{W}_{rev_i}|\mu, \Sigma) \cdot Pro(\mu, \Sigma))d\,\boldsymbol{W}_{rev_i} \end{aligned} \quad (3)$$

Because the overall rating is known, $\Psi = \{\,\boldsymbol{W}_{rev_1}, \ldots, \boldsymbol{W}_{rev_M}, \mu, \Sigma, \sigma^2\}$ contains the model parameters to be estimated by performing the maximum log-likelihood (ML) method. By identifying the optimal Ψ^* for maximizing the log-likelihood $\Psi^* = \arg\max_\Psi \sum_{(rev_i, p_j) \in S} \log Pro(R_{ij}|\Psi, r_{ij})$, we obtained the optimal values for $\boldsymbol{W}_{rev_i} (1 \le i \le M)$, which are the weighted preferences of reviewers REV on features $\mathcal{F}$, and will be further normalized into an unit vector.

[1]http://nlp.stanford.edu/software/lex-parser.shtml

4.2 Clustering Reviewers based on *LCRM* and Generating Product Recommendations

After inferring individual reviewer's weighted feature preferences from her/his written reviews, the reviewers' preference similarity network can be constructed where the edge denotes the preference similarity between a pair of reviewers. To exploit the value of such network, we have tried various approaches to generating recommendations. Particularly, we have attempted to further divide reviewers in this network into sub-communities where every sub-community contains a group of closely like-minded reviewers, as shown in Figure 4. We compared clustering-based approaches to the ones without clustering in the experiment, and found that the clustering can help increase the algorithm's accuracy (more details can be seen in Section 5). To cluster reviewers, we further tested two optional strategies: one employs the classic K-means technique; and the other adopts the Latent Class Regression Model (LCRM) since it can potentially more accurately learn the common interests among reviewers. In the following, we thus mainly describe the LCRM-based clustering and recommending method. The other implemented methods will be introduced in Section 5.2 ("Compared Methods").

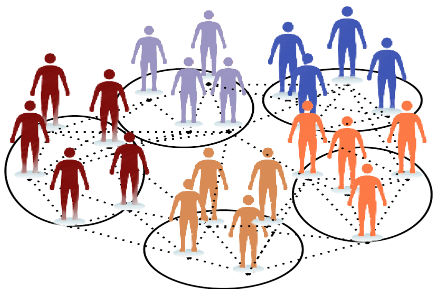

Figure 4: The reviewers' sub-communities in the implicit preference similarity network.

4.2.1 *Clustering Reviewers based on LCRM*

Latent class model (or called finite mixture model) is a popular statistical modeling technique which can be used for clustering. This model assumes the populations are drawn from a finite number of different distributions where each of the distribution is called as component. In [19], an extension is proposed to estimate a linear model for each component, which can be referred as *Latent Class Regression Model (LCRM)*. In the market segmentation area, the Latent Class Regression Model (LCRM) has been successfully applied to represent consumers' preference heterogeneity [26]. According to [26], LCRM has theoretical advantage over traditional clustering method (such as K-means) for conducting market segmentation, because it divides the users into clusters based on their membership probabilities (so that a user is assigned to a cluster only when this assignment has the highest probability) rather than relying on the pre-definition of distance threshold. However, it has been rarely investigated in the research field of recommender systems, in terms of its potential effect on enhancing the similarity measure among users.

In our case, since the major objective is to accurately identify a group of reviewers who could not only be closely similar to each other, but also well server for the matching with a new user according to their common preferences, we believe that LCRM could be more likely to retrieve the essentially like-minded reviewers by capturing their preference heterogeneity. Moreover, LCRM can simultaneously estimate cluster-level preferences along with the clusters, which can be used to explain the common preferences among the cluster of reviewers.

Specifically, assume that the population of reviewers can be divided into K clusters $\mathcal{C} = \{c_1, c_2, \ldots, c_K\}$. According to LCRM, if a reviewer rev_i belongs to the cluster c_k, the conditional probability of the overall rating R_{ij} could be defined as:

$$Pro(R_{ij}|\boldsymbol{X}_{ij}, c_k) = \mathcal{N}(R_{ij}|\,\boldsymbol{W}_{c_k}^T \boldsymbol{X}_{ij}, \sigma^2) \qquad (4)$$

where $\boldsymbol{W}_{c_k} \in \mathbb{R}^n$ is the cluster's weighted feature preferences (that represent the common preferences of this cluster of reviewers).

Because the overall rating that a reviewer assigned to a product is known, the above formula can be based to calculate the probability that a reviewer belongs to a cluster. Formally, a reviewer rev_i is placed in a cluster c_k if $q_k(rev_i) > q_h(rev_i)\ \forall c_k \neq c_h \in \mathcal{C}$ where

$$q_k(rev_i) = \prod_{(rev_i, p_j)\in\mathcal{S}} \frac{\pi_k \cdot Pro(R_{ij}|\boldsymbol{X}_{ij}, c_k)}{\sum_{c_h\in\mathcal{C}} \pi_h \cdot Pro(R_{ij}|\boldsymbol{X}_{ij}, c_h)} \qquad (5)$$

In the above formula, $q_k(rev_i)$ is the posterior probability that a reviewer rev_i belongs to a cluster c_k, and π_k is the prior probability. The full mixture likelihood can be accordingly defined as:

$$\mathcal{L}(\psi|\mathcal{S}) = \prod_{(rev_i, p_j)\in\mathcal{S}} \sum_{k=1}^{K} q_k(rev_i) \cdot Pro(R_{ij}|\boldsymbol{X}_{ij}, c_k) \qquad (6)$$

The parameter $\psi = \{\pi_1, \ldots, \pi_K, \boldsymbol{W}_{c_1}, \ldots, \boldsymbol{W}_{c_K}\}$ is estimated by the Expectation-Maximization (EM) algorithm, which seeks to identify the maximized log-likelihood by iteratively applying the following two steps:

- **Expectation step (E step)** which updates the posterior probability that a reviewer belongs to a certain cluster and derives the prior cluster probability:

$$\hat{\pi}_k = \frac{\sum_{i=1}^{M} q_k(rev_i)}{M}$$

- **Maximization step (M step)** which aims to find the optimal parameter values of $\hat{\psi}$ for maximizing Eq. 6:

$$\hat{\psi} = \max_{\psi} \mathcal{L}(\psi|\mathcal{S})$$

E- and M-steps are repeated until the Eq.6 converges. As a result, all reviewers are automatically classified into K disjoint clusters (K is set as 6 with the best recommendation accuracy through experimental trials).

4.2.2 *Generating Recommendations*

With the clusters of reviewers, it should be the time to generate appropriate product recommendations to a new user. Suppose through the preference elicitation method [6], the system can obtain the weighted feature preferences that a new user explicitly stated, which are represented as $\boldsymbol{W}_u = \{w_{f_i} | i \in \{1, 2, \ldots, n\}\}$. The value of w_{f_i} is set as zero if the corresponding feature weight is not explicitly stated by the user. Our target was then to retrieve the cluster of reviewers who are most relevant to the new user. Formally, the similarity between a cluster of reviewers and the new user is computed based on the Euclidean distance similarity:

$$sim(\boldsymbol{W}_u, \boldsymbol{W}_{c_k}) = \frac{1}{1+\sqrt{\sum_{i=1}^{n}(w_{f_i}(u) - w_{f_i}(c_k))^2}} \quad (7)$$

The reason why we chose Euclidean distance similarity measure is because it takes into account the magnitude in contrast with other similarity (e.g., cosine similarity). The new user is hence associated to the cluster with the highest similarity among others. Then, the products as reviewed by this cluster of reviewers are taken as recommendation candidates. Each product p_j is further computed with a prediction score:

$$PredictionScore(u, p_j) = \frac{\sum_{rev_i \in c_l \wedge (rev_i, p_j) \in \mathcal{S}} sim(\boldsymbol{W}_u, \boldsymbol{W}_{rev_i}) \times R_{ij}}{\sum_{rev_i \in c_l \wedge (rev_i, p_j) \in \mathcal{S}} sim(\boldsymbol{W}_u, \boldsymbol{W}_{rev_i})} \quad (8)$$

where c_l denotes the cluster of reviewers that is most relevant to the new user, R_{ij} is the overall rating that a reviewer gave to a product, and $sim(\boldsymbol{W}_u, \boldsymbol{W}_{rev_i})$ is the preference similarity between the new user u and the reviewer rev_i (which is formally calculated by replacing $\boldsymbol{W}_{c_k}$ with $\boldsymbol{W}_{rev_i}$ in Eq. 7).

Top-N products with higher prediction scores are finally returned to the new user as the recommendations (in our experiment, we tested the algorithm's performance when $N = 10, 20, 30$).

5. EXPERIMENT

5.1 Experiment Setup

The dataset as described in Section 3 was used to perform the experiment. Given that a new user has a target product to buy (which is her/his target choice), the experimental goal was to evaluate whether the target choice could be located in the recommendation list when being presented to her/him. For this goal, we concretely adopted the *leave-one-out* evaluation scheme [25]. That is, during each round, we excluded one reviewer from the dataset and performed testing on it. As a matter of fact, the excluded reviewer must satisfy two "new user" selection criteria, so that the product purchased by the reviewer can be taken as the new user's target choice when measuring the algorithm's recommendation accuracy: 1) s/he only commented one product, and 2) her/his overall rating on the purchased product is full marks (i.e., 5), indicating that s/he strongly likes the product. In our dataset, 1705 reviewers were found satisfying these criteria. Therefore, at a time, one of them was randomly chosen to behave as a new user. We further randomly selected subsets of the reviewer's full feature preferences (i.e., 40%, 60%, 80% and 100%) to represent the new user's various *preference completeness* levels.

5.2 Compared Methods

As mentioned before, we have developed several approaches to be compared to the LCRM-based reviewers' clustering and recommendation method (henceforth named as RS-LCRM). Specifically, four related approaches were implemented: one without the fusion of reviews, and other three with the fusion of reviews but by different means.

1. Recommending without the fusion of reviews (Baseline)

This is a baseline for us to verify the actual effect of reviews on enhancing the system's recommendation accuracy. It is purely based on the product's static features (i.e., technique specifications) for ranking. More specifically, given the new user's stated weighted preferences $\boldsymbol{W}_u$, the matching score $ProductScore(u, p_j)$ of each product p_j is defined as:

$$ProductScore(u, p_j) = \sum_{w_{f_l}(u) \in \boldsymbol{W}_u} w_{f_l}(u) \times s_{f_l}(p_j) \quad (9)$$

where p_j is the product, and $s_{f_l}(p_j)$ is the value of each static feature f_l ranging from 0.0 to 1.0. The values of static features were actually crawled from `www.buzzillions.com` and stored in the experiment dataset. The top-N products with higher scores are then included in the recommendation list.

2. Recommending with the fusion of reviews

The following three methods are all with the same inputs resulted from Section 4.1.2, which are the reviewers' weighted feature preferences as recovered from reviews, but they vary in the way of utilizing these inputs.

- *Review-fused baseline approach (RF-baseline)*

  This method primarily used reviewers' feature opinions to describe the products' feature space [1]. Concretely, there is a feature score computed for each feature of a product, by aggregating the feature's opinions derived from the product's reviews: $FeatureScore_{f_l}(p_j) = \frac{\sum_{(rev_i, p_j) \in \mathcal{S}} X_{ijl}}{m}$ where x_{ijl} denotes the feature f_l's opinion value in review rev_i to product p_j, and m denotes the number of reviews associated to the product p_j.

  Thus, the matching score of each product (according to the new user's preferences) is computed as:

  $$ProductScore(u, p_j) = \sum_{w_{f_l}(u) \in \boldsymbol{W}_u} w_{f_l}(u) \times FeatureScore_{f_l}(p_j) \quad (10)$$

  Hence, its difference from Eq. 9 is that it mainly relies on the feature's opinion value to determine its score.

- *Review-fused KNN approach (RF-k-NN)*

  We implemented this approach because it is a common way to integrate the similarity between user-reviewer into generating recommendations. That is, given the new user's current preference $\boldsymbol{W}_u$, it aims at first identifying a set of reviewers $\mathcal{K}$ who have similar feature preferences to the new user. The similarity between the new user and a reviewer is formally computed by replacing $\boldsymbol{W}_{c_k}$ with $\boldsymbol{W}_{rev_i}$ in Eq. 7.

  Then, a prediction score is assigned to each product p_j by following the basic collaborative filtering mechanism:

  $$PredictionScore(u, p_j) = \frac{\sum_{rev_i \in \mathcal{K}} sim(\boldsymbol{W}_u, \boldsymbol{W}_{rev_i}) \times R_{ij}}{\sum_{rev_i \in \mathcal{K}} sim(\boldsymbol{W}_u, \boldsymbol{W}_{rev_i})} \quad (11)$$

  where $|\mathcal{K}| = 2000$ through experimental trials. Still, top-N products with higher scores are recommended to the new user.

- *Review-fused K-means approach (RF-k-Means)*

  Another approach that we implemented is using the classic clustering technique, i.e., the K-means clustering, to divide the reviewers into K disjoint clusters $\{c_1, \ldots, c_K\}$ ($K = 6$), and then retrieve the cluster that is closest to the new user. Concretely, during conducting K-means clustering, a reviewer would be moved from one cluster to another if this process could minimize her/his squared distance from the cluster's centroid. The distance between a reviewer u_i and the centroid of c_k is defined as: $1/sim(\boldsymbol{W}_u, \boldsymbol{W}_{c_k\_centroid})$ (where $sim(\boldsymbol{W}_u, \boldsymbol{W}_{c_k\_centroid})$ is computed by replacing $\boldsymbol{W}_{c_k}$ with $\boldsymbol{W}_{c_k\_centroid}$ in Eq. 7).

  Then the new user's preferences are based to compute her/his distance from all clusters' centroids, and the cluster with the shortest distance is matched to the new user. Afterwards, the same recommendation procedure as in the LCRM-based approach is performed to generate recommendations.

Thus, it can be noticed that the main difference among *RF-k-NN, RF-k-Means, RF-LCRM*, exists at their processes of identifying similar reviewers to the new user. After the set of like-minded reviewers is located, the procedure of computing recommendations basically acts the same among them.

5.3 Evaluation Metrics

To choose appropriate evaluation metrics, we checked Kendall's tau, Hit-Ratio and Percentile, as they have been applied to measure the recommendation accuracy based on user preferences [12]. However, Kendall's tau was finally found improper because it assumes that each user prefers multiple items and targets to compute the similarity between the estimated preference ranking (over these items) and the "true preference ranking". But in our case, the "true preference ranking" is not available because each user normally only makes a single choice in the inexperienced, high-risk product domains. Thus, we decided to use Hit-Ratio and Percentile, as they can assess the algorithm's accuracy in including the "target choice" in the recommendation list:

- $H@N$ (Hit ratio @ top-N recommendations): it refers to the percent of successes that a new user's target choice appears in the top-N recommendation list.

$$H@N = \frac{\#\text{The number of successes within the top-N}}{\#\text{The total number of tested new users}} \tag{12}$$

- *Percentile*: it gives the percent of products which are ranked below the target choice [27].

$$Percentile = \frac{\sum_{t=1}^{T} \frac{|\mathcal{P}| - Rank_{target\_choice}}{|\mathcal{P}|}}{T} \tag{13}$$

  in which, $|\mathcal{P}|$ is the total number of products, and T is the number of tested new users.

5.4 Results Analysis

Table 2 shows the comparative results from the five approaches. First of all, it can be seen that all of the three user-reviewer similarity based methods, i.e., RF-k-NN, RF-k-Means, and RF-LCRM, perform much better than the two baseline methods, especially against RF-baseline that simply utilizes feature-opinion pairs to enrich products' feature space. As a matter of fact, no matter of how incomplete the new user's preferences are, the three methods that derived reviewers' weighted feature preferences from product reviews and further incorporated them to compute user-reviewer similarity, are more accurate than the others in terms of both hit ratio and percentile metrics. This finding hence distinguishes the impact of exploiting users' *weighted feature preferences* on increasing recommendation accuracy, especially in the inexperienced product domains where a reviewer is often associated to only one or few products.

Furthermore, the two clustering-based methods, RF-k-Means and RF-LCRM, are found behaving more accurate than RF-k-NN method at various preference completeness levels. This result implies that the pre-cluster of reviewers according to their feature preferences can be more likely to identify the like-minded reviewers to a new user, compared to the on-site retrieval of neighboring reviewers purely based on the new user's stated preferences. More notably, it is interesting to discover the outperforming accuracy of LCRM-based clustering method than K-means based, particularly when the new user's preferences are less complete. In fact, RF-LCRM achieves 0.229 and 0.261 hit ratios @ N=10 (plus 0.697 and 0.660 percentiles) respectively at 40% and 60% preference completeness levels, which are higher against all the other four methods. When the user's preferences become more complete (i.e., 80% and 100%), the K-means method obtains slightly higher hit ratio, but RF-LCRM still shows higher percentile (0.717) at 80%.

Table 2: Comparison of five algorithms' recommendation accuracy

New users' preferences	Method	Evaluation Metrics			
		H@10	*H*@20	*H*@30	*Percentile*
40% complete	Baseline	0.053	0.063	0.157	0.484
	RF-baseline	0.059	0.059	0.170	0.502
	RF-k-NN	0.146	0.198	0.324	0.692
	RF-k-Means	0.188	0.193	0.335	0.592
	RF-LCRM	**0.229**	**0.245**	**0.409**	**0.697**
60% complete	Baseline	0.059	0.065	0.157	0.486
	RF-baseline	0.048	0.058	0.177	0.694
	RF-k-NN	0.196	0.201	0.334	0.690
	RF-k-Means	0.234	0.234	0.385	0.643
	RF-LCRM	**0.261**	**0.269**	**0.456**	**0.660**
80% complete	Baseline	0.071	0.101	0.162	0.492
	RF-baseline	0.054	0.064	0.178	0.513
	RF-k-NN	0.186	0.213	0.345	0.700
	RF-k-Means	0.270	0.281	0.416	0.690
	RF-LCRM	0.247	0.254	**0.428**	**0.717**
100% complete	Baseline	0.080	0.120	0.145	0.486
	RF-baseline	0.043	0.043	0.175	0.518
	RF-k-NN	0.206	0.261	0.345	0.703
	RF-k-Means	0.300	0.312	0.469	0.710
	RF-LCRM	0.242	0.253	0.423	0.706

6. CONCLUSIONS

In conclusion, this paper presented a novel recommendation framework for aiding new buyers' decision making in inexperienced product domains. Particularly, it aimed at constructing product reviewers' preference similarity network through recovering their weighted feature preferences, and utilizing this implicit network to generate recommendation. We also proposed a Latent Class Regression Model (LCRM) based method to divide reviewers into sub-communities, each of which contains reviewers with tight preference-closeness. We then matched the new user to these clusters to identify the most relevant one. Products as reviewed by the relevant reviewers were finally taken as recommendation candidates. Our experimental results show the effectiveness of basing reviewers' weighted feature preferences for enhancing recommendation accuracy. More notably, the LCRM-based method behaved more accurate than the four compared methods (including the K-means based clustering method), especially when the new user's stated preferences were incomplete. It hence suggests that this method can be effectively combined with our previously proposed critiquing agents [8], for which the critiquing agent is responsible for eliciting the new user's feature preferences, and the LCRM-based method can take charge of generating recommendations to the user particularly when the elicited preferences are less complete.

7. REFERENCES

[1] S. Aciar, D. Zhang, S. Simoff, and J. Debenham. Informed recommender: Basing recommendations on consumer product reviews. *IEEE Intelligent Systems*, 22:39–47, May 2007.

[2] G. Adomavicius and Y. Kwon. New recommendation techniques for multicriteria rating systems. *IEEE Intelligent Systems*, 22:48–55, May 2007.

[3] M. Balabanović and Y. Shoham. Fab: content-based, collaborative recommendation. *Commun. ACM*, 40:66–72, March 1997.

[4] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proc. UAI'98*, volume 461, page 43ĺC52, 1998.

[5] P. Chatterjee. Online Reviews - Do Consumers Use Them? *ACR 2001 Proceedings*, pages 129–134, 2001.

[6] L. Chen and P. Pu. Survey of preference elicitation methods. In *Technical Report No. IC/200467*, Lausanne, Switzerland, 2004.

[7] L. Chen and P. Pu. Evaluating critiquing-based recommencler agents. In *Proc. AAAI '06*, volume 1, pages 157–162, 2006.

[8] L. Chen and P. Pu. Hybrid critiquing-based recommender systems. In *Proc. IUI '07*, pages 22–31. ACM, 2007.

[9] W. Desarbo, M. Wedel, M. Vriens, and V. Ramaswamy. Latent class metric conjoint analysis. *Marketing Letters*, 3:273–288, 1992.

[10] A. Esuli and F. Sebastiani. Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proc. LRECąŕ06*, pages 417–422, 2006.

[11] C. Fellbaum. *WordNet: An Electronic Lexical Database*. The MIT Press, Cambridge, MA, 1998.

[12] A. Gunawardana and G. Shani. A survey of accuracy evaluation metrics of recommendation tasks. *J. Mach. Learn. Res.*, 10:2935–2962, Dec. 2009.

[13] M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proc. KDD '04*, pages 168–177, New York, NY, USA, 2004. ACM.

[14] N. Jakob, S. H. Weber, M. C. Müller, and I. Gurevych. Beyond the stars: exploiting free-text user reviews to improve the accuracy of movie recommendations. In *Proc. TSA '09*, pages 57–64, New York, NY, USA, 2009. ACM.

[15] R. L. Keeney, H. Raiffa, and D. W. Rajala. Decisions with multiple objectives: Preferences and value trade-offs. *Systems, Man and Cybernetics, IEEE Transactions on*, 9(7):403, July 1979.

[16] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42:30–37, August 2009.

[17] K. Lakiotaki, N. F. Matsatsinis, and A. Tsoukias. Multicriteria user modeling in recommender systems. *IEEE Intelligent Systems*, 26:64–76, March 2011.

[18] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. 5th Berkeley Symp. on Math. Statist. and Prob.*, volume 1, pages 281–297. University of California Press, 1967.

[19] P. McCullagh and J. A. Nelder. *Generalized Linear Models*. Chapman & Hall, London, 2nd edition, 1989.

[20] R. D. N. Sahoo, R. Krishnan and J. P. Callan. Collaborative filtering with multi-component rating for recommender systems. In *Proc. 6th WITS*, Milwaukee, WI, USA, 2006.

[21] J. Payne, J. Bettman, and E. Johnson. *The Adaptive Decision Maker*. Cambridge University Press, May 1993.

[22] D. Poirier, I. Tellier, R. Fessant, and J. Schluth. Towards text-based recommendations. In *Proc. RIAO '10*, pages 136–137, Paris, France, France, 2010.

[23] A. M. Popescu and O. Etzioni. Extracting product features and opinions from reviews. In *Proc. HLT '05*, pages 339–346, Stroudsburg, PA, USA, 2005.

[24] L. Qi and L. Chen. Comparison of model-based learning methods for feature-level opinion mining. *Proc. WI '11*, 1:265–273, 2011.

[25] B. Smyth, L. McGinty, J. Reilly, and K. McCarthy. Compound critiques for conversational recommender systems. In *Proc. WI '04*, WI '04, pages 145–151, Washington, DC, USA, 2004.

[26] M. Wedel and W. A. Kamakura. *Market Segmentation - Conceptual and Methodological Foundations*, volume 9. 2000.

[27] A. Yates, J. Joseph, A. M. Popescu, A. D. Cohn, and N. Sillick. Shopsmart: product recommendations through technical specifications and user reviews. In *Proc. CIKM '08*, pages 1501–1502, New York, NY, USA, 2008. ACM.

[28] J. Yu, Z. J. Zha, M. Wang, and T. S. Chua. Aspect ranking: identifying important product aspects from online consumer reviews. In *Proc. HLT '11*, pages 1496–1505, Stroudsburg, PA, USA, 2011.

[29] W. Zhang, G. Ding, L. Chen, C. Li, , and C. Zhang. Generating virtual ratings from chinese reviews to fuse into collaborating filtering algorithms. *ACM TIST*, 2012.